UNSW ICPC Workshop T3W1
Implementation Problem Set
Source: South Pacific Divisionals (2018)

1. Register an account at https://prog4fun.csse.canterbury.ac.nz/
2. Go to https://prog4fun.csse.canterbury.ac.nz/mod/quiz/view.php?id=51
3. Try to solve and implement the problems in this document
    - It's recommended that you attempt them in the given order (Flow, Easiest Problem, God's Number)
    - You can code in the browser or on your computer
    - Ask/message someone if you need help – understanding what to do and getting your first submission through can be tricky when starting out

Problem F
Flow
Time limit: 2 seconds

Implementation Problem 1

(you can skip the first two paragraphs)

Maximus Aquius, the Roman aqueduct themed supervillain, is trying to irrigate the streets of Rome. He plans to do this by building a network of aqueducts that direct water from the surrounding countryside to the city. There are a number of water sources around Rome, each producing a fixed volume of water each hour. Each source of water has some altitude, and water can only flow downhill. If a water source has a lower altitude than the city itself, it cannot be used to irrigate the city.

Maximus Aquius, having no faith in your ability to solve complex problems, has hired a more talented computer scientist to compute the set of additional aqueducts he needs to build. Having now decided on his plan, he needs to figure out the logistics necessary to build the network.

Your task is to determine how many labourers must be hired to construct the aqueducts to ensure they are completed before Caesar returns from Gaul. Each labourer can build exactly 1 cubit of aqueduct each day. Multiple labourers can work on the same aqueduct at the same time and multiple aqueducts can be worked on at the same time, but a worker can only work on a single aqueduct on any given day. Given the length of each of the aqueducts (in cubits), what is the minimum number of labourers that must be employed to build all aqueducts before Caesar returns?

Input

The first line of input contains two integers m ($1 \leq m \leq 100\,000$), which is the number of aqueducts to be built, and n ($1 \leq n \leq 100\,000$), which is the number of days until Caesar returns.

The next m lines describe the aqueducts. Each line contains a single integer ℓ ($1 \leq \ell \leq 1\,000$), which is the length of this aqueduct in cubits.

Output

Display the minimum number of labourers required.

Sample Input 1

```
5 10
2
2
2
2
2
```

Sample Output 1

```
1
```

Sample Input 2

```
6 16
18
3
5
4
1
1
```

Sample Output 2

```
2
```

Implementation Problem 2

Problem E
Easiest Problem
Time limit: 1 second

When doing ICPC contests, one of the most important skills is correctly determining the difficulty of each problem quickly. To get the best placement in an ICPC contest, it is in a team's best interest to solve the easiest problem first and work on the harder problems later.

To ensure that they find the easiest problem, all three members of Team Merlin will read all of the problems and rank each one in terms of difficulty. Each ranking is an integer between 1 and 100, where 1 is the easiest and 100 is the hardest. A problem's *difficulty score* is the average of the three team members' rankings. The problem with the lowest difficulty score is deemed to be the *easiest problem*. In the case of a tie, the problem that received the lower ranking from the team captain is the easiest problem. It is guaranteed that the captain will not give two problems the same rank.

What is the easiest problem? Sample 2 is not an indication of actual difficulty.

Input

The first line of the input contains a single integer P ($1 \leq P \leq 15$), which is the number of problems in the contest.

The next P lines describe the problems. Each line starts with a string, which is the name of the problem. Then follow three integers a, b, c ($1 \leq a, b, c \leq 100$), which are the three rankings of the members of Team Merlin. The team captain's ranking is a. The problem's name uses only lowercase and uppercase letters and consists of between 1 and 50 characters inclusive.

The P problem names will be distinct. Problem names are case-sensitive.

Output

Display the name of the easiest problem.

Sample Input 1

```
2
AnEasyProblem 5 3 7
AHardProblem 92 81 97
```

Sample Output 1

```
AnEasyProblem
```

Sample Input 2

```
12
AustralianVsAmerican 10 10 10
BombsAhoy 20 20 20
CrypticClues 25 25 20
DubiousRecording 35 23 15
EasiestProblem 1 1 1
Flow 30 30 30
GodsNumber 70 75 80
Holiday 34 15 88
IslandOfLove 100 100 100
JuiceMachine 50 12 32
KrazyTaxi 23 30 30
LoveActually 99 99 99
```

Sample Output 2

```
EasiestProblem
```

Implementation Problem 3

Problem G
God's Number
Time limit: 1 second

The Rubik's Cube is a $3 \times 3 \times 3$ cube puzzle. There are six colours and when solved, the nine squares on each face have the same colour. See Figure G.1.

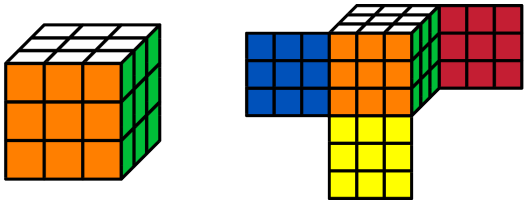
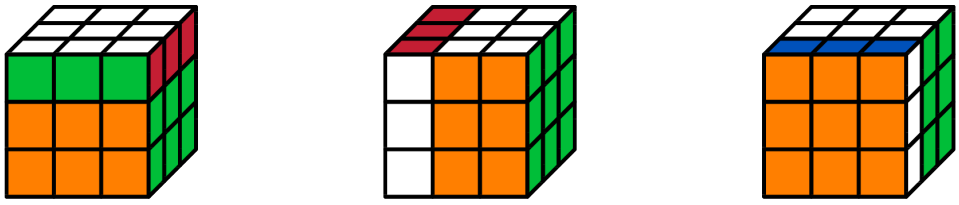


Figure G.1: A solved Rubik's Cube.

Each of the six faces can be rotated either clockwise or counterclockwise, which rotates the colours on that face as well as alters one row on the four neighbouring faces. Here are the six clockwise rotations by 90°:



`U` (Up) `L` (Left) `F` (Front)

`D` (Down) `R` (Right) `B` (Back)

In 2010, it was proven that any Rubik's Cube configuration can be solved in at most 20 moves (one *move* here is rotating one face by either 90°, 180° or 270° clockwise). In 2014, it was proven that if you were only allowed rotations of 90° or 270° (clockwise), then any Rubik's Cube configuration can be solved in at most 26 moves.

Jacob is attempting to tackle the final case: only allowing rotations of 90° (clockwise). He has been working towards a solution for several days, but cannot seem to even solve the Rubik's Cube! Thankfully, he started with a solved Rubik's Cube and he has kept track of the moves that he has made so far. Given this information, help solve the Rubik's Cube for him using only rotations of 90° (clockwise). You *do not* need to solve the Rubik's Cube in the minimum number of moves.

Input

The input consists of a single line containing a single string, which is the sequence of moves that Jacob has performed on the Rubik's Cube. The string consists of only `B`, `D`, `F`, `L`, `R` and `U`. The Rubik's Cube starts solved, then the sequence of moves are applied, in order. The length of the string is between 1 and 50, inclusive.

Output

Display a sequence of moves that solves the Rubik's Cube. Your solution does not need to be optimal. The sequence of moves must be in the same format as the input. Your sequence can be of any length between 1 and 200, inclusive.

Sample Input 1	Sample Output 1
DUDUDUD	U

Sample Input 2	Sample Output 2
FF	FF

Sample Input 3	Sample Output 3
URRFBRBBRUULBBRUUUDDDRRFRRRLBBUUFF	URRFBRBBRUULBBRUUUDDDRRFRRRLBBUUFF

Sample Input 4	Sample Output 4
FFFF	LLLL

Sample Input 5	Sample Output 5
FB	FFFBBB